UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/803,514	03/08/2001	Sridhar Obilisetty	026507-000300US	6110

20350 7590 06/17/2008

TOWNSEND AND TOWNSEND AND CREW, LLP
TWO EMBARCADERO CENTER
EIGHTH FLOOR
SAN FRANCISCO, CA 94111-3834

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2193	

MAIL DATE	DELIVERY MODE
06/17/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	09/803,514	OBILISETTY, SRIDHAR	
	Examiner	Art Unit	
	Tuan A. Vu	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *13 March 2008*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☐ Claim(s) *1-6, 8-25* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-6 and 8-25* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 3/13/2008.

As indicated in Applicant's response, no claims have been amended. Claims 1-6, 8-25 are pending in the office action.

Claim Objections

2. Claims 1, 13, 25 are objected to because of the following informalities: 'text files executing in combination according to said program'. Based on the lack of teaching from the claim regarding how a text files operates like a computer process or code instruction, and the definition of 'text files' and 'executing' in the Disclosure (*XML syntax* - pg. 8; *process of a computer* - pg. 13), the phrase 'text files executing in combination' is deemed an inappropriate use of language because, as commonly accepted, a Web-based interpreter normally interprets XML data and a program instruction is executed with a runtime engine (i.e. the engine executes the instruction). The text nature of even a XML syntax is interpreted by a corresponding engine, but a text file (or combination thereof) does not execute. The term 'executing in combination' as recited is not given any weight. Appropriate correction is required.

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-6, 8-12, and 25 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bloch et al., USPN: 2002/0129129 (hereinafter Bloch).

As per claim 1, Bloch discloses a method for implementing an application on a client computer system, said method comprising:

receiving at said client a plurality of text files; each defining a component of the application (e.g. Fig. 1-3; steps 66, 68, 70 - Fig. 5; Fig. 6; pg. 10-11, para 0095, 0096);

executing a program resident on said client system for using a combination of said text files to create an application (e.g. *AVM 221* - Fig. 2; para 0068-0069, pg. 8; *GUI frame ... waits for the user* - para 0057-0058, pg. 6 – Note: installed and iniatialized AVM -- para 0037, pg. 4; para 0050, pg. 5; Fig. 2-3 – by means of text script files and providing GUI rendered components for user events to take place reads on executing program resident on client); and

creating said application on said client system according to said program (e.g. Fig. 2, 5; para 0044, pg. 4), wherein said application is executable independent from said program (e.g. *Client Tasks, Host tasks* – para 0086-87, pg. 10; *once ... initialized ... user interact ...visible frame ... next user* interaction … *System handler 315 to exit* - para 00100, pg. 11; *message box ... tasks ... unique to the operating system* – para 0102, pg. 11; *remote procedures* – para 0103, pg. 11; *Database handler ... user to access data ... Database* – para 0105, pg. 12; *user interface ... accessing any database ... to test new software* – para 0109; para 0062-0063, pg. 7 – Note: user-driven actions within specific OS browser application (Fig. 9-10) including invoking of methods, system tasks or executing remote calls for testing of components which are assembled and compiled within the AVM non-application-specific framework – see para 0037, pg. 4; para 0044, pg. 4 – reads on application *executable* dependent on user and **contextually independent** from the neutral AVM context – see para 0050, pg. 5; 0057-0058, pg. 6 - which is only setting a development framework).

But Bloch does not explicitly disclose checking automatically for updated versions of said text files. Bloch however addresses the urge for providing latest set of files in accordance to appropriate version of script file or virtual machine files with regard to version (e.g. para 0051-0053, pg. 5-6); hence has taught checking of files and their latest upgrade; and further discloses parsing XML or HTTP text files and browser (Fig. 5; para 0030, pg. 3). It is noted that a file version being included in the header of XML or HTTP files is implicitly disclosed owing to known browser/markup language technologies at the time the invention was made. In light of the desirability of updating browser files to meet the appropriate virtual machine or execution environment version as mentioned above along with the implicitly automated version/format compliance checking when markup files are processed by a browser engine, it would have been obvious for one of ordinary skill in the art at the time the invention was made to enhance the desire for version checking as shown by Bloch so that using the browser engine for automatically checking the text files for the latest upgraded file version because this would enable the executing environment to be provided with the appropriate files according to the version expected for such environment as addressed above.

As per claim 2, Bloch discloses XML format (Fig. 1, 2, 4,5).

As per claim 3, Bloch discloses server central source for managing and distributing applications or modifications for applications (e.g. *upgrades, fixes* - pg. 3, para 0032; pg. 5, para 0045-0046; pg. 12, para 0109).

As per claim 4, refer to claim 3 and Bloch's Fig. 1, 2, 4,5.

As per claim 5, Bloch discloses executing an application, sending a request and executing the application in parallel while waiting for response from the request (e.g. … *reports*

to the Application Handler 302, … periodically updates -- pg. 9, para 0080 – 0082 – Note: resolving a URL with data retrieval while leaving the GUI window on for being updated on tree events changes and notified of download status is equivalent to executing application while waiting for remote response)

As per claim 6, Bloch discloses connectionless application execution (e.g. pg. 8, para 0069; pg. 12, para 0108)

As per claim 8, Bloch discloses modifying application by using a newer text files replacing older files (*upgrades, fixes* - pg. 3, para 0032; *most recent ...version* - pg. 12, para 0107).

As per claim 9, Bloch discloses graphical user interface (e.g. Fig. 6).

As per claim 10, Bloch discloses application being communication preferences for database invocation (e.g. pg. 7, para 0063; Preference Handler 303 - Fig. 4)

As per claim 11, Bloch discloses data management application (e.g. step *52* – Fig. 5; Manager 301 -Fig. 4 - Note: downloading files to assemble manager module reads on application being a management application).

As per claim 12, Bloch discloses component being part of logic of application (pg. 1, para 0012; pg. 4, para 0037).

As per claim 25, Bloch discloses a computer-readable medium having program code on a computer system to perform a method comprising:

installing a plurality of text files, each defining a component of the application (e.g. Fig. 1; steps 66, 68, 70 - Fig. 5; Fig. 6; pg. 10-11, para 0095, 0096; Fig. 2-3);

installing a program wherein said program comprises instructions for using a combination of said text files to create an application (e.g. *AVM 221* - Fig. 2; para 0037, pg. 4; para 0050, pg. 5; Fig. 2-3 Note: installed and iniatialized AVM -- para 0037, pg. 4; para 0050, pg. 5; Fig. 2-3 – by means of text script files and providing GUI rendered components for user events to take place reads on executing program resident on client); and

creating said application on said client system according to said program (e.g. Fig. 2, 5);

wherein said application is executable independent from said program (e.g. *Client Tasks, Host tasks* – para 0086-87, pg. 10; *once ... initialized ... user interact ...visible frame ... next user* interaction … *System handler 315 to exit* - para 00100, pg. 11; *message box ... tasks ... unique to the operating system* – para 0102, pg. 11; *remote procedures* – para 0103, pg. 11; *Database handler ... user to access data ... Database* – para 0105, pg. 12; *user interface ... accessing any database ... to test new software* – para 0109; para 0062-0063, pg. 7 – Note: user-driven actions within specific OS browser application (Fig. 9-10) including invoking of methods, system tasks or executing remote calls for testing of components which are assembled and compiled within the AVM non-application-specific framework – see para 0037, pg. 4; para 0044, pg. 4 – reads on application *executable* dependent on user and **contextually independent** from the neutral AVM context – see para 0050, pg. 5; 0057-0058, pg. 6 - which is only setting a development framework).

receiving automatically any updated versions of said execution environment files (e.g. *to make sure ... most current versions ... download and install automatically*-- para 0051-0053, pg. 5-6) in response to version checking.

But Bloch does not explicitly disclose automatically receiving any updated versions of said text files in response to version checking. Bloch teaches database storage for download support for version upgrades and the implicit version checking by browsers of markup files as set forth in claim 1. Thus, it would have been obvious for one of ordinary skill in the art at the time the invention was made to enhance Bloch deployment and XML processing environment so that not only script or virtual machine files, but also the text files such as the XML files are checked for update and automatic re-download, according to the version checking as known in browser technologies set forth in claim 1, because of the desirability to conform not only application files but also specification files, a concept inherent to browsers using XML metadata without which format and version conformance would potentially create application execution conflicts.

5. Claims 13-24 are rejected under 35 U.S.C. 103(a) as being unpatentable over Bloch et al., USPN: 2002/0129129 (hereinafter Bloch), in view of Landsman et al., USPN: 6,314,451 (hereinafter Landsman).

As per claim 13, Bloch discloses a computer system with bus, processor coupled to a bus (*Client PC* - Fig. 1; pg. 12, para 0108) for implementing an application comprising the steps:

receiving (text files);

executing (program resident);

creating (application), wherein said application is executable independent from said program (refer to claims 1, 25). All these steps limitations have been addressed in claim 1; hence are rejected herein with the corresponding rejections as set forth therein, respectively.

But Bloch does not disclose uploading results from using said application to a server computer system. However, Bloch teaches legacy browsers (para 0048) so as to obviate

redeploying using alternate means as well as remote persisting of records on reuseable user application data until the user decide to change the application preferences (para 0069-0070, pg. 8). Landsman teaches a browser environment where the user can customize application-related preferences by providing mouse-clicking interface analogous to the user-driven method of Bloch. Retransmitting of deployed application results are evidenced further in Landsman's method wherein logs of application execution data are uploaded to a server (e.g. col. 31, line 62-67). Based on the desirability to persist user preferences and the implied benefits of legacy of schema being used for users as mentioned above, it would have been obvious for one of ordinary skill in the art at the time the invention was made to enhance Bloch server database records so that there is an uploading of application results as taught by Landsman so that improvement of previous results or user schema preferences would lend some insight as implied via Bloch's teachings or via the analysis of logs data by Landsman.

As per claim 19, Bloch discloses text files particular to client system (e.g. pg. 4, para 0037; pg. 5, para 0047, 0050)

As per claims 14-18, 20-24, these claims correspond to claims 2-6, 8-12 respectively, hence are rejected with the corresponding rejections as set forth therein, respectively.

Response to Arguments

6. Applicant's arguments filed 3/13/08 have been fully considered but they are not persuasive. Following are the Examiner's observation in regard thereto.

35 USC § 103 Rejection:

(A) Applicants have submitted that the downloaded application is not executable independent from the AVM as cited (Appl. Rmrks pg. 8, 3rd para). Suppose that the argument made it clear

that Bloch's cited paragraphs are compared against the language of claim 1 (which is not evident) following is how the claim language has been addressed. The claim recites 'executing a program … on said computer system … the program comprises instructions for using … text files to create said application'. The downloaded package and installed (para 0050, pg. 5) in Bloch is to address the program resident on the client system being executed, because when installation has finished activation this package, this package becomes the Virtual Machine AVM; as an application being readily activated, it has capability for interpreting text files (instructions for using a combination of text files) in order to help the user's development of logic thereby implementing the user's target application (see para 0044, pg. 4, Fig. 1); that is, this capacity reads on 'for using a combination of ... text files to create said application'. Nowhere in Block is shown that **by** installing and instantiating this AVM machine now residing at the client machine, this finished stage of installation and the readyAVM tool made available (by way of 'executing a program resident') affects the intent and purposes of the developer in a fashion that enforce a necessary inter-dependency when user actions are taken. In regard to 'executable independent from said program', the cited portions in the Office Action describe user's driven actions including executing scripts, method calls, remote access which as a whole amount to what is interpreted from the recited 'said application' being executable; that is, prior to being executable, the AVM is the application for developing software, based on XML, and on a browser which is started (see para 0050, para 0057; Fig. 9-10; Fig. 2-3) as a result of installing and initializing ("executing a program resident") a GUI settings/development interface from a downloaded AVM package and/or from a core *starter module*. In the state where this application is executable, every actions by the user are considered application dynamically effectuated by

user's actions and being unrelated to the setting up of the AVM environment; that is, via these user actions the development application is executable independent of the 'executing a program resident' as analyzed above. The claim does not describe 'application' in specific details to enforce what 'application' actually consists of; nor does the claim language enforce that 'application' is the final code formed from the text files and in its compiled form, is actually executed (emphasis added) in an runtime environment which is not same as that of the 'program resident'. The 'independent from said program' is deemed fulfilled by the rejection.

For the sake of argument, let's assume that the AVM application by Bloch when made available to the user is analogized to 'said application' and that installing this AVM is analogized to 'executing a program resident'. When this 'application' is recited as 'executable', it is not understood (i) whether application is intended for future runtime on different platforms or (ii) it is the framework instance itself that can be executed or under execution; nor are there details that (iii) would enforce a particular and actually effectuated execution instance that would clearly **distinguish** Bloch's user's utilizing of this application - for example Bloch's AVM framework -- **from** this AVM initial installation/settings (see Bloch: para 0050), which are deemed a terminated process; or that (iv) would preclude running features (by the developer in Bloch) of this instantiated AVM (see para 0050) from being unrelated to a previous OS' installation context of Bloch's AVM. The static framework (AVM) being activated for use amounts to application created from text files (see Fig. 2-3); and the user action in invoking calls amounts to the dynamic aspect of this application being *executable*.

The argument is not providing facts that would sufficient deemed convincing to overcome the rejection regarding this 'independent from' limitation.

(B) The Applicants has submitted that tasklist in para 0086 cannot be executable both by the AVM and independent therefrom (Appl. Rmrks pg. 8, bottom). The tasklist is executable within the user's context, and is deemed totally un-related to the runtime of the OS whereby the received AVM is first unpackaged then activated for the user to subsequently go about running the feature therein or scripts and compile code, which are dependent only of the user's context and preference. This argument is not convincing for the main reasons set forth in section A.

(C) The Applicants has submitted that in paragraph 0100, AVM session ends after an execution to emphasize that dependency between executed applications and AVM (Appl. Rmrks pg. 9, top). The rejection has not treated the program ('executing the program resident') as a activation process whereby the installed AVM framework/GUI tool is made available. Figure 11 addresses case where additional data were to be fetched after the AVM is installed. The script executing and requesting a system call to exit a given session has no dependency whatsoever with the already-completed AVM activation runtime ('program resident') whereby the AVM tool has been made available for user to add more data. Moreover, the dynamic nature of system call and handled-related session cannot be equated to a stable state of a Virtual Machine graphical environment. The argument is not convincing.

(D) The Applicants have submitted that message boxes and OS calls cannot be created by the downloaded files (Appl. Rmrks pg. 9, 2nd para). The user's driven actions contribute to the executable application that are based on interpreted script and XML files (see Fig. 1-5). The user actions carrying out the executable functionality of the AVM application is deemed independent from the terminated process of establishing the AVM graphical framework. Applicant's arguments fail to comply with 37 CFR 1.111(b) because they amount to a general

allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the reference. The very textual constructs of the pointed claim language has been analyzed and mapped with proper cited portions of Bloch; and Applicant's analogizing AVM as being the 'program resident' and OS calls as 'application' is not deemed commensurate with the analysis as set forth above. The user's layer event that solely depends on the user amounts to this user context of 'application' when this application is 'executable' because 'executable' entails that some calls can or cannot be invoked solely under the control of a human being. The Applicants argued the 'procedure calls' are not application executable being created from text files (Appl. Rmrks pg. 9, bottom). The established GUI framework refers to a application that is statically available after the parsing has been performed (see Fig 2-3) as executing this process of creating the GUI has ended (refer to section A); the dynamic evolving of this static environment is construed as actions executed by the user, making the user's remote calls a component of said dynamic state of the AVM framework; and this is independent of the installed Gui framework, necessarily independent of the runtime whereby this GUI tool is made available.

(E) Likewise, Applicants have submitted that database handler, and accessing database are not applications created from text files (Appl. Rmrks, pg. 10, top). This improper interpretation of the Office Action is referred back to sections A, D above.

(F) Applicants have expressly admitted that the Specifications cannot be read into the language claimed as 'executable independent from said program' (Appl. Rmrks pg. 10, 4th para). Granted that this admission holds true, which can potentially trigger some USC 112 type of analysis, the claim subject matter cannot be viewed as though in a vacuum, and is yet to be

examined based on merits that necessitate interpretation by one of ordinary skill. Thus, this language can be interpreted in two manners: first, in light of the Specifications, then based on commonly accepted meaning. At the time of prosecution, 'executable independent' has been readily interpretable at face value. Section A has analyzed how 'executable' has been understood, and based on this, the claims have been addressed, thereby obviating the urge to bring teachings from the Specifications into the above process of Examiner's interpretation.

In whole, the arguments are deemed insufficient as to put forth clear and distinguishing details between Bloch and the language of the claims; further, attacking one separate reference in light of a USC § 103 rejection would not be a prima facie or correct approach to overcome this type of rejection; the claims will stand as set forth in the rejection.

Conclusion

7. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (571) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Lewis Bullock can be reached on (571)272-3759.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence - please consult Examiner before using) or 571-273-8300 (for official correspondence) or redirected to customer service at 571-272-3609.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.
Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

/Tuan A Vu/

Primary Examiner, Art Unit 2193

May 29, 2008